

## IMPALA: matching a protein sequence against a collection of PSI-BLAST-constructed position-specific score matrices

Alejandro A. Schäffer<sup>1</sup>, Yuri I. Wolf<sup>1</sup>, Chris P. Ponting<sup>1,3</sup>, Eugene V. Koonin<sup>1</sup>, L. Aravind<sup>2</sup> and Stephen F. Altschul<sup>1</sup>

<sup>1</sup>National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, USA and <sup>2</sup>Department of Biology, Texas A&M University, Biological Sciences Building West, College Station, TX 77843, USA

Received on March 19, 1999 ; revised on July 28, 1999; accepted on August 4, 1999

### Abstract

**Motivation:** Many studies have shown that database searches using position-specific score matrices (PSSMs) or profiles as queries are more effective at identifying distant protein relationships than are searches that use simple sequences as queries. One popular program for constructing a PSSM and comparing it with a database of sequences is Position-Specific Iterated BLAST (PSI-BLAST).

**Results:** This paper describes a new software package, IMPALA, designed for the complementary procedure of comparing a single query sequence with a database of PSI-BLAST-generated PSSMs. We illustrate the use of IMPALA to search a database of PSSMs for protein folds, and one for protein domains involved in signal transduction. IMPALA's sensitivity to distant biological relationships is very similar to that of PSI-BLAST. However, IMPALA employs a more refined analysis of statistical significance and, unlike PSI-BLAST, guarantees the output of the optimal local alignment by using the rigorous Smith–Waterman algorithm. Also, it is considerably faster when run with a large database of PSSMs than is BLAST or PSI-BLAST when run against the complete non-redundant protein database.

**Availability:** The IMPALA source code, the wolf1187 database, and the aravind105 database are freely available from the NCBI ftp site [ncbi.nlm.nih.gov](ftp://ncbi.nlm.nih.gov). The databases may be found in the subdirectory <ftp://ncbi.nlm.nih.gov/pub/impala>. The source code is in [ftp://ncbi.nlm.nih.gov/toolbox/ncbi\\_tools](ftp://ncbi.nlm.nih.gov/toolbox/ncbi_tools). Some IMPALA executables for different implementations of UNIX are in <ftp://ncbi.nlm.nih.gov/blast/executables>. IMPALA

has been added as a search option on the Blocks Database Server (<http://blocks.fhcrc.org/blocks/impala.html>) using a library of PSSMs derived from the BLOCKS database.

**Contact:** [schaffer@helix.nih.gov](mailto:schaffer@helix.nih.gov)

### Introduction

This paper describes a new software package, IMPALA (Integrating Matrix Profiles And Local Alignments), for searching a database of position-specific score matrices (PSSMs) using a protein sequence as query. It complements the popular programs BLAST and Position-Specific Iterated BLAST (PSI-BLAST) (Altschul *et al.*, 1997) that search a database of proteins sequences using, respectively, a simple sequence and a PSSM as query. Position-specific score matrices, sometimes called profiles and sometimes encoded as hidden Markov models (HMMs), have a long history (McLachlan, 1983; Staden, 1984; Schneider *et al.*, 1986; Taylor, 1986; Berg and von Hippel, 1987; Dodd and Egan, 1987; Gribskov *et al.*, 1987; Patthy, 1987; Stormo and Hartzell, 1989; Gribskov *et al.*, 1990; Brown *et al.*, 1993; Lawrence *et al.*, 1993; Tanaka *et al.*, 1993; Baldi *et al.*, 1994; Tatusov *et al.*, 1994; Yi and Lander, 1994; Durbin *et al.*, 1998). The primary intuition is that a multiple alignment of related proteins can reveal position-specific residue propensities which, properly deployed, should increase the sensitivity with which distant homologs are recognized. For example, in an alignment position where leucine is completely conserved, a leucine might receive a high positive score, whereas in a position where a variety of hydrophobic residues have been observed, leucine might receive a lower but still positive score.

An effective system for searching a database of PSSMs has three elements, each of which poses separate challenges: (1) the construction and curation of multiple

<sup>3</sup>Present address: MRC Functional Genetics Unit, Department of Human Anatomy and Genetics, University of Oxford, South Parks Road, Oxford OX1 3QX, UK.

alignments for many sets of interesting proteins; (2) the conversion of the multiple alignments into position-specific scoring systems; (3) an algorithm for searching a database of position-specific models.

Some important approaches to problem 1 include: PROSITE (Hofmann *et al.*, 1999), the BLOCKS database (Henikoff and Henikoff, 1994), Pfam (Bateman *et al.*, 1999; Sonnhammer *et al.*, 1997), and SMART (Ponting *et al.*, 1999b). PROSITE addresses problems 2 and 3 with the software pftools (Bucher *et al.*, 1996). The BLOCKS project has addressed these problems with various multiple alignment software tools in the Blocks Database Server (Henikoff *et al.*, 1999). IMPALA has recently been added as a search option on the Blocks Database Server (<http://blocks.fhcrc.org/blocks/impala.html>) using a library of PSSMs derived from the BLOCKS database. The Pfam and SMART projects use HMMs as encoded in the package HMMER (Eddy, 1996). HMMs relevant to problems 2 and 3 were introduced recently by several groups (Brown *et al.*, 1993; Tanaka *et al.*, 1993; Baldi and Chauvin, 1994; Baldi *et al.*, 1994; Krogh *et al.*, 1994) to model large protein families such as globins. Their theory is described in detail in the book (Durbin *et al.*, 1998), and Eddy (1998) gives a review of relevant HMM literature. There have been various other related approaches to these problems as well (Tatusov *et al.*, 1994; Neuwald *et al.*, 1997; Park *et al.*, 1998).

These search tools are widely used, and each is more effective at identifying distant similarities than position-independent methods. Despite the many successes of position-specific methods, position-independent methods such as FASTA (Pearson and Lipman, 1988) and BLAST (Altschul *et al.*, 1990; Gish and States, 1993; Altschul *et al.*, 1997) remain more popular for protein similarity searching.

BLAST took an important step towards position-specific searching by adding the PSI-BLAST module (Altschul *et al.*, 1997). PSI-BLAST addresses the multiple alignment construction problem by coalescing the pairwise matches found by BLAST into a multiple alignment. It then converts this alignment into a PSSM by calculating scores for each position as the logarithm of ratios between predicted and background residue frequencies. The ways in which PSI-BLAST confronts problems 1 and 2 thus are quick and convenient for researchers who already use BLASTP. Although it is possible to start PSI-BLAST directly with a PSSM built in a previous run, this does not address problem 3 because each PSI-BLAST run uses only a single scoring matrix. The ability of PSI-BLAST to systematically detect subtle but structurally and functionally relevant relationships between proteins has been demonstrated by several groups (Mushegian *et al.*, 1997; Altschul and Koonin, 1998; Huynen *et al.*, 1998; Park *et al.*, 1998; Rychlewski *et al.*, 1998; Aravind and

Koonin, 1999; Teichmann *et al.*, 1999; Wolf *et al.*, 1999).

IMPALA complements PSI-BLAST by allowing users to systematically compare a single query sequence with a database of PSSMs of the form constructed by PSI-BLAST. For the purposes of rapid code development and testing, IMPALA has been written as a set of separate programs, but it is integrated into the National Center for Biotechnology Information (NCBI) software toolkit and reuses much of the code and theory of PSI-BLAST.

IMPALA's local alignment scores are provably optimal, unlike BLAST's or PSI-BLAST's, because IMPALA uses the Smith–Waterman algorithm (Smith and Waterman, 1981; Gotoh, 1982). This algorithm is much slower than the BLAST algorithm for searching the same database of sequences. However, a PSSM database that attempts to avoid redundancy will be a fraction of the length of the underlying sequence database used to construct its models. For PSSM and standard sequence databases required to match a similar range of sequences, the PSSM database should be able to be substantially shorter.

We believe that with further development of well-curated PSSM libraries, IMPALA and similar tools should become an important addition to the existing repertoire of database-searching methods.

## Methods

The IMPALA package consists of three programs: makemat, copymat, and impala. The programs makemat and copymat are auxiliary programs that manipulate the PSSM database; they need to be run only once per database, much like the formatdb auxiliary program is run to create a BLAST-searchable sequence database representation. To search a query against a formatted PSSM database the user runs just the impala program. We use upper case IMPALA to refer to the entire package, and lower case impala to refer to the main program.

The Methods section is divided into three parts. First, we describe the representation of PSSMs in IMPALA and two preprocessing programs that manipulate that representation. Second, we describe the theory used to compute alignments and *E*-values. Third, we describe the engineering of the main program impala.

### *PSSM representation and preprocessing*

A PSSM *M* generated by PSI-BLAST is associated with a sequence whose residues define the positions, and are used for displaying pairwise alignments. The command-line version of PSI-BLAST allows the user to store the current PSSM in a file with the -C (stands for 'checkpoint') option, and to restart from a previously stored checkpoint file with the -R (stands for 'restart') option. For a sequence of length *L*, the checkpoint is stored as an  $L \times 20$  array of floating point ratios of the general form

$q_{ij}/p_j$ , where  $q_{ij}$  represents the predicted frequency of residue  $j$  in position  $i$ , and  $p_j$  represents the background frequency of residue  $j$ .

PSI-BLAST checkpoints are stored as byte-encoded frequency ratios to insure maximum flexibility in scaling. The makemat auxiliary program converts these ratios into integral ASCII score matrices, so that PSSM sets can be examined and freely distributed in a machine-independent fashion. While constructing these matrices, makemat also records in an auxiliary file certain useful information such as the statistical parameters for each matrix, gap costs, and the overall size of the PSSM set.

The program copymat converts the matrix files generated by makemat into one-large byte-encoded integer matrix of dimensions  $D \times 20$ , where  $D$  is the number of positions in the PSSM database. This conversion allows impala to read the entire PSSM database with a single mmap command, rather than fetching individual matrices with multiple disk accesses.

For each PSSM library, the preprocessor programs makemat and copymat are each run once, converting it to the format used by the main program impala.

#### *Alignments, scores and E-values in impala: theory and gaps*

The impala program seeks an optimal gapped local alignment of the query sequence  $Q$  against each PSSM  $M$ . Each PSSM has an associated sequence which is used only as a placeholder, so that pairwise alignments can be displayed in BLAST format. Optimal local alignments are guaranteed by using the Smith and Waterman (1981) algorithm, extended for affine gap costs (Gotoh, 1982). Only alignments with an  $E$ -value below some user-controlled threshold are reported. If  $Q$  and  $M$  have a local alignment with  $E$ -value below this threshold, impala seeks additional disjoint local alignments using a multiple-match extension (Waterman and Eggert, 1987) of the Smith and Waterman (1981) algorithm.

Alignment scores are converted to  $E$ -values using empirical parameters for the extreme value distribution (Altschul and Gish, 1996; Altschul *et al.*, 1997). We explain in some detail how this is done because impala's approach departs slightly from that of various versions of BLAST (Altschul *et al.*, 1990, 1997), and as a result may generate more accurate  $E$ -values.

An analytic theory allows one to estimate the statistical significance of ungapped local alignments of two sequences  $A$  and  $B$  (Karlin and Altschul, 1990; Dembo *et al.*, 1994). The theory applies in the asymptotic limit of long sequences, but a short sequence 'edge-correction' has been proposed (Altschul and Gish, 1996). In brief, given an ungapped local alignment with score  $S$ , its  $E$ -value (the expected number of distinct alignments from the comparison of random sequences that would achieve a score at

least  $S$ ) can be estimated by:

- (1) Compute parameters  $\lambda$  and  $K$  that depend on the residue composition of  $A$ ,  $B$  and on the residue substitution scores used.
- (2) Compute adjusted lengths  $l_A$  and  $l_B$  of  $A$  and  $B$  that take into account that high-scoring alignments cannot start near the end of either sequence.
- (3) Compute a normalized score  $S' = \lambda S - \ln(K)$ .
- (4) Compute the  $E$ -value as  $l_A l_B e^{-S'}$ .

When sequence  $B$  is part of a database, to correct for multiple comparisons the term  $l_B$  is replaced in step 4 by the sum of the adjusted lengths of all database sequences (Altschul *et al.*, 1994). This correction, used in all BLAST programs, reflects the a priori assumption that database sequences are likely to yield biologically meaningful local alignments in proportion to their length. An alternative view, implemented in some sequence analysis programs, e.g. FASTA (Pearson and Lipman, 1988), is that all database sequences are a priori equally likely to yield a meaningful match, regardless of their lengths. Taking this view, one would instead multiply the term in step 4 by the number of sequences in the database.

If  $A$  is the query sequence and  $B$  is a database sequence, then the original ungapped BLAST algorithm (Altschul *et al.*, 1990) used the actual residue composition of  $A$  at step 1, but an average residue composition for  $B$ . Call the resulting scale parameter  $\lambda_{av}$ , and the parameter that would be obtained using the true composition of  $B$ ,  $\lambda_{tr}$ . If  $\lambda_{av} > \lambda_{tr}$ , then the  $E$ -value will be underestimated, possibly leading to false positives; conversely, if  $\lambda_{tr} > \lambda_{av}$ , the  $E$ -value will be overestimated, leading to false negatives. Relatively small differences between  $\lambda_{tr}$  and  $\lambda_{av}$  can lead to large differences in  $E$ -value estimates because of the exponentiation at step 4. We found that in practice  $\lambda_{av}$  is usually greater than the values of  $\lambda_{tr}$ , so that one may underestimate but is unlikely to overestimate  $E$ -values by assuming an average composition for database sequences. If a database sequence has a residue composition that leads to low values of  $\lambda_{tr}$ , it does not necessarily contain isolated regions of restricted residue composition than can be removed by filtering (Wootton and Federhen, 1993; Altschul *et al.*, 1994).

Using IMPALA with the proteome of *Mycoplasma genitalium* and the PSSM database wolf1187, as reported below, there were 1426 potential matches with a score high enough to merit testing the ratio  $\lambda_{tr}/\lambda_{av}$ . The ratio was  $> 1.0$  only 4% of the time, with a maximum value of 1.10. In contrast, the ratio was  $< 0.9$  31% of the time, with a minimum value of 0.56.

There is no proof that the theory above applies to gapped alignments, but many computational experiments suggest that it does (Smith *et al.*, 1985; Collins *et al.*, 1988; Mott,

1992; Waterman and Vingron, 1994; Altschul and Gish, 1996; Pearson, 1998). A difficulty is that the statistical parameters  $\lambda$  and  $K$  can not be derived analytically, but must be estimated by simulation with random or real but unrelated sequences. Some database search programs such as FASTA (Pearson and Lipman, 1988) generate as a byproduct scores for many unrelated sequences, and these can be used to estimate the statistical parameters (Pearson, 1998). Gapped BLAST (Altschul *et al.*, 1997) can not use the same strategy, because it generates scores for only a small number of unrelated sequences. Accordingly, it uses random simulation to estimate ahead of time the statistical parameters  $\lambda_g$  and  $K_g$  ('g' stands for gapped) for any particular gapped alignment scoring system. One disadvantage of this approach is that such estimates use a standard residue composition, which does not reflect the composition of either the query or database sequence.

PSI-BLAST generates at each iteration a new PSSM, and it is impossible to anticipate the form of these matrices for the purpose of random simulation. Because it would be too time-consuming to re-estimate a gapped  $\lambda$  for each new PSSM, Altschul *et al.* (1997) propose a scaling strategy. Generalizing the formulas of Karlin and Altschul (1990) to PSSMs in the obvious way, they calculate ungapped  $\lambda$  for the comparison of a given PSSM to a sequence of average composition. By scaling the PSSM, this parameter can be made equal to  $\lambda_u$  for ungapped alignments of protein sequences with standard composition using a standard substitution matrix. It is then postulated that if gaps are permitted, the parameter  $\lambda$  for PSSM-sequence comparisons using a particular set of gap costs will be the same as the previously estimated  $\lambda_g$  for sequence-sequence comparisons. This hypothesis is supported reasonably well in practice (Altschul *et al.*, 1997).

PSI-BLAST scales a PSSM just once, assuming the proteins in the database can be described by an average residue composition. IMPALA could use an analogous strategy, by assuming the query has a standard residue composition, and adopting the same PSSM scale as used by PSI-BLAST. We found, however, that although this works well in most cases, there is an occasional query-PSSM pair whose true ungapped scale parameter  $\lambda_{utr}$  ('utr' for ungapped, true) is much smaller than  $\lambda_u$ . This leads to great overestimates of  $\lambda_{gtr}$  by  $\lambda_g$ , and attendant exaggeration of the significance of alignments involving the corresponding query-PSSM pair. To address this problem, for each query-PSSM that appears to produce a significant alignment, we rescale the PSSM based upon the actual residue composition of the query, and rerun the pairwise comparison. The PSSM rescaling is described in steps 4 and 7 below. The intervening steps 5 and 6 assess whether a query-PSSM pair can achieve a reportable  $E$ -value before actually scaling the matrix at step 7. This

produces quite accurate  $E$ -values, as will be seen below. The same strategy could be adopted by both PSI-BLAST and gapped BLAST (Altschul *et al.*, 1997) to yield more accurate  $E$ -values at the cost of a slight decrease in speed.

The next subsection describes how the above theory is engineered into the main program *impala*.

### *Impala main loop*

The main loop of *impala* performs the following steps to find the first match of a query  $Q$  to a PSSM  $M$ . For better precision, *impala* uses by default, at steps 1–12, matrix scores that are 100 times as large as those used in BLAST. That is, the underlying floating point log-odds ratios are first multiplied by 100 and then rounded. The main loop uses two thresholds  $T_1 \geq T_2$  for deciding which  $E$ -values merit further computational effort; by default,  $T_1$  is chosen as  $5T_2$ .

- (1) Let `firstScore`  $\leftarrow$  the Smith–Waterman score for aligning  $Q$  to  $M$ .
- (2) Convert `firstScore` to `firstEvalue` using  $\lambda_g$ .
- (3) If `firstEvalue`  $> T_1$ , then stop.
- (4) Compute  $\lambda_{utr}$ , and define  $F = \lambda_{utr}/\lambda_u$ , which, as explained above, is the factor used to rescale  $M$ .
- (5) If  $F > 1$ , proceed to step 7, skipping step 6. Otherwise, multiply `firstScore` by  $F$  and use that estimated score for the rescaled matrix to compute a `secondEvalue`.
- (6) If `secondEvalue`  $> T_2$  stop.
- (7) Rescale  $M$  so that it has ungapped  $\lambda$  equal to  $\lambda_u$ , and call it  $M_{tr}$ . This is done by multiplying each entry in a floating point representation of  $M$  by  $F$  and then rounding to the nearest integer.
- (8) Let `thirdScore`  $\leftarrow$  the Smith–Waterman score for aligning  $Q$  to  $M_{tr}$ , and record the last position pair of an optimal alignment.
- (9) Convert `thirdScore` to `thirdEvalue`.
- (10) If `thirdEvalue`  $> T_2$ , then stop.
- (11) Compute the other endpoint of an optimal alignment.
- (12) Compute the optimal alignment as described below.
- (13) Prepare the optimal alignment for display.

When a query  $Q$  has a match with  $M_{tr}$  of  $E$ -value  $< T_2$ , steps 8–13 of the above procedure are iterated to find additional disjoint but significant local alignments (Waterman and Eggert, 1987).

After computing optimal alignment scores and endpoints with the Smith–Waterman algorithm (step 11), *impala* uses the X-drop alignment method of Zhang *et al.*

(1998) to compute alignments. This is the same alignment procedure used in BLASTP, and it finds alignments where no subinterval of aligned residues and gap characters contributes a partial score  $< -X$ , for some positive threshold  $X$ . Using  $X$ -drop alignments allows *impala* to be easily integrated with the NCBI toolkit; e.g. if in the future, some decision is taken to change how BLAST pairwise alignments are displayed, the decision will automatically be implemented in *impala*. Alignments produced by  $X$ -drop are not necessarily optimal for any specific value of  $X$ . Since *impala* computes the score of the optimal alignment before finding the  $X$ -drop alignment, it can use the following well-known trick to convert the Monte Carlo (likely to give the optimal answer in one try)  $X$ -drop algorithm into a Las Vegas (guaranteed to give the optimal answer after an uncertain number of tries) algorithm:

```

optScore ← Smith–Waterman score for  $Q, M_{tr}$ 
 $X \leftarrow$  default initial value
do {
    testAlign ←  $X$ -drop alignment of  $Q, M_{tr}$ 
    testScore ← score of testAlign
     $X \leftarrow 2X$ 
} while testScore < optScore

```

This method increases  $X$  until the  $X$ -drop method finds an optimal alignment. The line  $X \leftarrow 2X$  can be replaced by any strictly monotone increasing integer-valued function of  $X$ . Run-time profiling (in the computer science sense of this term) demonstrated that well over 90% of *impala*'s run time is spent rejecting PSSMs that give a low score. Therefore, the number of times the `do_while` loop is iterated has little effect, and most any choice of initial  $X > 0$  and function for incrementing  $X$  will run quickly enough in practice.

At output step 13, *impala* divides the alignment scores by 100 to bring them back to the standard scale used by BLAST, so that users are not misled by seemingly high scores. This makes the *impala* output score for matching  $Q$  to  $M_{tr}$  nearly equal the score produced by PSI-BLAST with query  $M$  matched against  $Q$  in the database. There may be small score differences due to the effects of better scaling and rounding in *impala*. The  $E$ -values may be quite different. The alignments should be nearly the same, except that the roles of 'Query' and 'Subject' are reversed.

## Two profile databases

To measure the accuracy of IMPALA output, it is convenient to use databases of PSSMs for which homology predictions have been made directly using PSI-BLAST, and these predictions thoroughly evaluated. We thus tested IMPALA on two PSSM databases created for other projects. The most we can hope is for IMPALA to be comparable to PSI-BLAST in prediction accuracy, but

faster than even a single BLAST search.

The first database we used consists of a collection of 1187 PSSMs (Wolf *et al.*, 1999), hereafter called *wolf1187*, that correspond to the folds in the SCOP database (Hubbard *et al.*, 1999; Murzin *et al.*, 1995). In the original study, 1195 PSI-BLAST PSSMs were constructed to represent most members of the SCOP 1.35 classification (Wolf *et al.*, 1999). Of the over 400 folds present in SCOP, 284 were used to generate the collection of PSSMs (the remaining folds were excluded for reasons detailed in Wolf *et al.*, 1999), so the correspondence between PSSMs and folds is many-to-one. There were two minor differences between *wolf1187* and the original version of this collection (Wolf *et al.*, 1999). In Wolf *et al.* (1999) two PSSMs were dropped because they represented folds seen only in viruses. Here we retain these PSSMs, but drop eight that are in effect position-specific instantiations of the underlying BLOSUM62 matrix (Henikoff and Henikoff, 1992), arising because PSI-BLAST found no significant matches to the original queries.

Wolf *et al.* (1999) evaluated the part of their PSSM database corresponding to the 30 most common folds on 15 complete, or nearly complete proteomes. For each sequence in each proteome, an initial homology prediction was recorded if the best PSSM match for the sequence gave an  $E$ -value, based on the size of the non-redundant protein sequence database (*nr*), below 0.01. To establish true and false positives, these predictions were then examined for the conservation of motifs typical of the respective protein families and superfamilies. By running PSI-BLAST in the opposite direction, i.e. by comparing the protein sequences encoded in the analyzed genomes to *nr*, registering hits to proteins present in PDB and, again, testing for the presence of diagnostic motifs, some false negatives were identified.

We believe that the evaluation of predictions for the *M.genitalium* proteome (Fraser *et al.*, 1995) is nearly perfect, since all the protein sequences encoded in this small genome were examined in detail, and especially because the fold assignments have been largely corroborated by four independent studies (Fischer and Eisenberg, 1997; Huynen *et al.*, 1998; Rychlewski *et al.*, 1998; Teichmann *et al.*, 1998). Although the analysis of the 467 sequences in the *M.genitalium* proteome described in print (Wolf *et al.*, 1999) included only PSSMs representing the 30 most common folds, similar evaluation was subsequently performed for all PSSMs in this collection (Y.I.Wolf and E.V.Koonin, unpublished work; <ftp://ncbi.nlm.nih.gov/pub/koonin/FOLDS/index.html>).

The second database we used consisted of 105 PSSMs and was originally constructed for the purpose of comparing the relative abundance of a set of widespread regulatory and signal-transduction domains in the proteomes of the yeast *Saccharomyces cerevisiae* and the nematode

worm *Caenorhabditis elegans*; the performance of this library, hereafter aravind105, was evaluated essentially as described for wolf1187 (Chervitz *et al.*, 1998).

We tested aravind105 on the *S.cerevisiae* proteome, for which a set of true positive matches was expertly curated. In short, for the protein domain families that are part of this PSSM collection, true positives in yeast were established by a series of transitive searches with PSI-BLAST initiated with different seed sequences, followed by a case-by-case analysis of the alignments generated. To confirm individual predictions, these alignments were checked for characteristic features of the relevant protein families. In this test we considered all IMPALA matches for each query; in the wolf1187 test we considered only the top match. The main reason for this difference is that most *M.genitalium* proteins have a single protein domain, and thus correspond to just one fold, whereas a single eukaryotic signalling protein often possesses more than one domain from aravind105.

## Results

We evaluate IMPALA on sensitivity in finding homologs while avoiding false positives, accuracy of reported *E*-values, and speed. Since IMPALA is targeted to users of BLASTP and PSI-BLAST, the evaluation methods are focused on those programs.

### Sensitivity

To measure sensitivity, we need a large set of queries and a database of PSSMs between which all true positives (homologies) are known.

The set of 467 proteins of the parasitic bacterium *M.genitalium* (Fraser *et al.*, 1995) matched against the wolf1187 database, and the set of 6406 (as of February 28, 1999) proteins of *S.cerevisiae* matched to the aravind105 database, should satisfy these criteria, as explained above. In Wolf *et al.* (1999) the searches were done by running PSI-BLAST with the representative sequence of each PSSM against a database of *M.genitalium* proteins, and *E*-values were calculated based on the size of the nr database at the time (approx.  $7 \times 10^7$  residues). A 'fold prediction' was registered when a reported match had an *E*-value  $\leq 0.01$ . Only the most significant match for each *M.genitalium* protein was recorded as a prediction.

By analogy with the above test, we ran IMPALA with each *M.genitalium* query against the wolf1187 PSSM database. A 'prediction' was registered when the most significant match had an *E*-value below some cutoff. A cutoff of 0.01, based on the total size of wolf1187, makes sense in a semi-automated setting where predictions will be checked further by other methods, and it is desirable to minimize false negatives. We also used a cutoff of  $3 \times 10^{-5}$  ( $= 0.01$  multiplied by the ratio of wolf1187 size/old nr

**Table 1.** Number of matches in a performance comparison of IMPALA and PSI-BLAST using wolf1187 PSSMs and *M. genitalium* queries

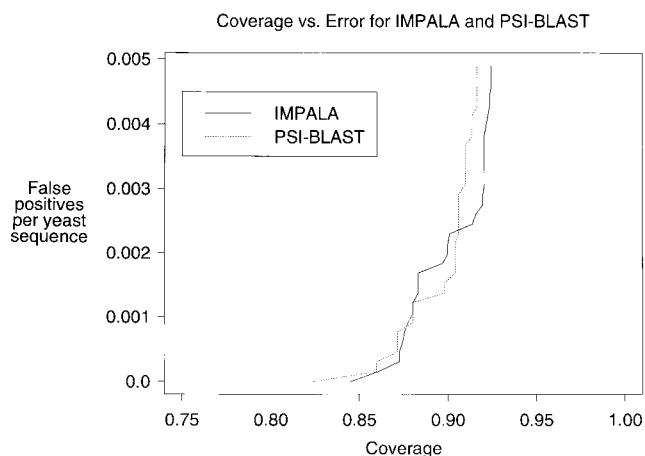
	IMPALA	IMPALA	PSI-BLAST
<i>E</i> -value threshold	0.01	$3 \times 10^{-5}$	0.01
True positives	204	192	196
False positives	8	2	4
False negatives	23	35	31

size) that is more appropriate in a fully automatic setting where it is desirable to minimize false positives. The results are shown in Table 1.

With the 0.01 cutoff, IMPALA finds slightly more true positives as well as false positives than does PSI-BLAST. With the  $3 \times 10^{-5}$  cutoff, IMPALA finds slightly fewer true and false positives than does PSI-BLAST. This comparable sensitivity is quite acceptable because of the gain in efficiency. To classify one new protein, one needs only a single IMPALA query, while one needs 1187 PSI-BLAST queries. The two false positives in the middle column of Table 1 have *E*-values  $< 10^{-20}$ , suggesting these matches arise from flaws in the way PSI-BLAST constructs PSSMs. This is of course a danger when using a fully automated approach like PSI-BLAST to construct PSSMs. In contrast, SMART (Ponting *et al.*, 1999b), as well as aravind105, are based on careful curation of multiple alignments, but this limits the number of profile models that can be built quickly.

For the test of aravind105 and *S.cerevisiae*, we started with a list of 998 true positive matches derived from detailed, case-by-case analyses. Since the true positives were compiled primarily using PSI-BLAST, we would expect PSI-BLAST's performance to be very good on this test set. This test was also more stringent because here we evaluated not just the top match, but all IMPALA matches with an *E*-value below a specified threshold.

A useful way to study the relative sensitivities of IMPALA and PSI-BLAST is by plotting coverage (i.e. percent of true positives found) vs. false positives per yeast sequence (see e.g. Brenner *et al.*, 1998). Advantages of such a plot are that it elides the question of the two comparisons' differing search spaces, and also that of the relative accuracy of the two methods' reported *E*-values. However, it requires at least a few 10s of false positives to get a meaningful plot. The results for the aravind105 vs. *S.cerevisiae* test are shown as a coverage vs. false positives plot in Figure 1. To compute coverage we assumed 1012 true positives: the 998 found originally, plus 14 newly uncovered by IMPALA and confirmed by careful transitive searches with different starting queries. Again, when equal numbers of false positives are allowed, IMPALA and PSI-BLAST yield quite similar coverage.

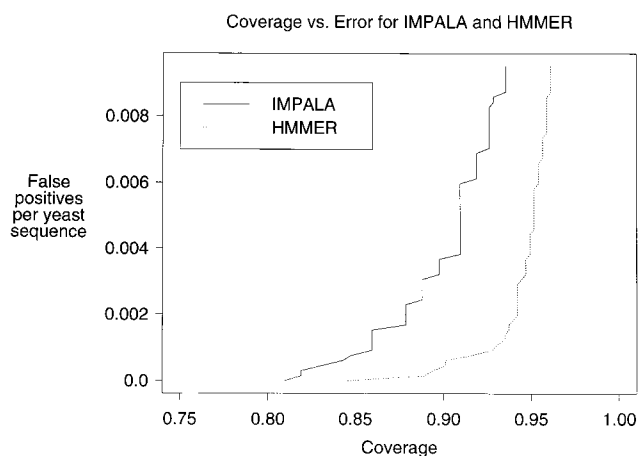


**Fig. 1.** Plot of coverage vs. false positives per yeast sequence for IMPALA and PSI-BLAST using *S.cerevisiae* sequences and aravind105 PSSMs. For each program, matches were sorted in increasing order of *E*-value, and the points on the plot represent the tradeoffs between fraction of true positives and fraction of false positives at different *E*-value cutoffs.

We were further interested in comparing the performance of IMPALA to that of other PSSM-based methods, starting from equivalent points. In part because of the results of the next subsection, it appeared the most stringent test would be to compare IMPALA to HMMER, which works best using a set of multiple alignments as its starting point. This comparison is difficult to design since a principal virtue of IMPALA is that it uses PSSMs generated on the fly by PSI-BLAST from master–slave rather than true multiple alignments, and it operates on these PSSMs rather than a database of multiple alignments. However, PSI-BLAST can convert master–slave multiple alignments to PSSMs, thereby permitting a comparison of IMPALA with programs that use a multiple alignment as the input description of a profile.

Specifically, the performances of IMPALA and HMMER (version 2.1; S.Eddy, unpublished, <http://hmmer.wustl.edu/>) were compared in detecting intracellular signalling domains represented within the complete set of *S.cerevisiae* proteins. For this purpose 45 multiple alignments of domains were obtained from the SMART database (Ponting *et al.*, 1999b). HMMs were calculated for each of these 45 alignments using HMMER's *hmm-build* and *hmmcalibrate* modules, and default parameters. Each yeast sequence was compared with the HMM library using the *hmmmpfam* program.

PSI-BLAST derives PSSMs by discarding insertions, relative to a 'master' query sequence. Thus, for comparison with HMMER, multiple PSI-BLAST-style PSSMs were derived from each multiple alignment, and used



**Fig. 2.** Plot of coverage vs. false positives per yeast sequence for IMPALA and HMMER using *S.cerevisiae* queries and 45 curated multiple alignments. The multiple alignments are converted to HMMs for HMMER and sets of PSSMs for IMPALA.

as a database for IMPALA. For this multiple PSSM construction, all sequences represented in an alignment were clustered using a single-linkage algorithm on the basis of sequence-to-sequence BLAST scores. The clustering threshold was automatically optimized within the preset range (0.1–1.0 bits/position), to obtain numbers of clusters close to 10. One sequence from each cluster was arbitrarily chosen as the master, and PSSMs were created using the *-B* option of PSI-BLAST. The resulting number of PSSMs for each multiple alignment ranged from 1 to 40.

Yeast sequences that gave high-scoring hits to the 45 SMART domain families were classified as either true positives (homologs) or false positives (unlikely homologs). The assignments drew upon the manually curated homology assessments embodied within SMART, the analysis of signaling domains published in Ponting *et al.* (1999a), other information from the literature, and additional PSI-BLAST database searches.

The results of this analysis are shown in Figure 2, using a coverage vs. error plot, as before. In this test, HMMER outperformed IMPALA in coverage by a few percentage points over the complete range of false-positive rates. This reveals limitations of PSI-BLAST-constructed PSSMs, compared with HMMs. Furthermore, we observed that IMPALA performed poorly and not comparably with HMMER in detecting domains that consist of multiple repeats; IMPALA systematically underestimated the number of repeats, although the presence of at least one copy was detected (data not shown).

The performance of HMMER critically depends on the quality of the multiple alignments it is provided. The

alignments used in this test were manually curated and optimized for performance with HMMER (Ponting *et al.*, 1999b). Thus we believe that the test in Figure 2 is a conservative estimate of IMPALA's performance. Generating large sets of well-curated multiple alignments is laborious and time-consuming. Therefore the principal use of the current implementation of IMPALA will be with large libraries of PSI-BLAST-constructed PSSMs, such as wolf1187, which may not be accompanied by the corresponding multiple alignment libraries. IMPALA's performance with such relatively crude PSSMs should be only slightly inferior to that of HMMER, one of the most powerful HMM-based methods employing carefully optimized multiple alignments.

#### Accuracy of reported *E*-values

To evaluate the accuracy of the *E*-values reported by IMPALA, we submitted as queries either shuffled or reversed protein sequences, and recorded how many matches fell below various thresholds. For simplicity, we again used the 467 *M.genitalium* proteins and the wolf1187 PSSM database. Shuffled queries mimic completely random sequences that preserve only the composition bias of true proteins. Reversed queries preserve some of the neighborhood structure of actual proteins, and may yield a more appropriate test of statistical significance. Table 2 shows that in both tests IMPALA gives roughly the expected number of matches at various thresholds, with reversed queries tending to yield slightly more matches than shuffled queries. IMPALA's *E*-values appear somewhat conservative: 47 total matches are expected to have an *E*-value < 0.1, and 467 to have an *E*-value < 1.0, but 34 and 318 are observed, respectively, in the reversed query test.

As explained above, IMPALA rescales the score matrix for each match. We found that if this is not done, six profiles out of the 1187 give many matches to either reversed or shuffled queries. This results from compositional bias in some query sequences that is systematically favored by the PSSM scores in these six cases. *E*-values are distorted in the same way when these protein-PSSM pairs are compared in PSI-BLAST. The consequences are distributed over many queries in IMPALA, but concentrated on a small number of queries in PSI-BLAST, because the problematic PSSMs in IMPALA correspond to queries in PSI-BLAST. Therefore, PSI-BLAST *E*-value tests (as in Altschul *et al.*, 1997) may miss or poorly estimate the severity of the problem, depending upon how often the queries used lead to PSSMs where  $\lambda_{\text{utr}}$  may be much smaller than  $\lambda_u$ .

We performed statistical tests using the same queries on a number of other software packages that can search profile libraries. These tests do not measure the ability of various packages to separate true relationships from

chance similarities, but simply the reliability of the *E*-values (*p*-values for BLIMPS) they report. These tests are easier to do than a sensitivity comparison, but are still difficult because of different profile libraries and extremely different notions about how to parameterize the programs. To give each program the best chance possible in this test, we used the default profile library most associated with the program. It would be better to compare all programs on the same library, but this is not feasible because of the four packages we tested, only MAST (Bailey and Gribskov, 1998) is comparable to IMPALA in allowing one to take as input a query and a library of PSSMs.

As shown in Table 2, the *E*-values returned by the hmmpfam program of HMMER (Eddy, 1996) using the Pfam 4.0 (Bateman *et al.*, 1999) library are roughly equivalent to IMPALA's. The *p*-values returned by BLIMPS (Wallace and Henikoff, 1992) using the BLOCKS database (Henikoff and Henikoff, 1994) are optimistic by a factor greater than 3 at  $p = 0.1$ , and greater than 10 at  $p = 0.001$ . The *E*-values returned by MAST (Bailey and Gribskov, 1998) using the PSSMs in the wolf1187 library are generally unreliable. The *E*-values reported by the program pfscan (Bucher *et al.*, 1996) using the PROSITE-derived library provided with the code are optimistic by a factor of 2–4 depending on query type and *E*-value threshold. The method for calculating *p*-values in BLIMPS is undergoing refinement (J.Henikoff, personal communication). Also, MAST appears designed to operate primarily on profiles of the sort output by the companion program MEME (Bailey and Elkan, 1995), rather than the full-length PSSMs used in this test, and may produce more accurate *E*-values in that context.

#### Speed

Because IMPALA performs a Smith–Waterman alignment of the query to each PSSM in its database, it is substantially slower than BLAST when the two programs are executed on databases of equivalent length. However, IMPALA usually will be run on a PSSM database much smaller than the protein database typically used by BLAST. Therefore, to compare the speed of IMPALA and BLAST in a meaningful way, we tested IMPALA on the large wolf1187 PSSM database (256 703 total matrix positions), and BLAST on the nr protein database (108 411 201 total residues). Table 3 shows that in this context (i.e. a database length ratio of about 420), IMPALA runs more than three times as fast as BLASTP on a spectrum of query lengths. We selected protein queries with length near a multiple of 100 to help clarify how running time depends on query length. For the timing test we ran only the main program impala, because the two preprocessor programs need to be used only once per database. The timing experiment was run on one 168 MHz

**Table 2.** Number of matches using reversed (Rev.) and permuted (Perm.) *M.genitalium* queries. For IMPALA and MAST we used the wolf1187 PSSM library; for HMMER we used the hmmpfam program and the Pfam 4.0 library of HMMs; for BLIMPS we used the built-in BLOCKS database; for pfscan we used the PROSITE-derived library provided with the code. BLIMPS reports *p*-values, not *E*-values, but these are essentially equivalent over the range in question. NR stands for not reported

<i>E</i> -value threshold	IMPALA		HMMER		BLIMPS		MAST		pfscan	
	Rev.	Perm.	Rev.	Perm.	Rev.	Perm.	Rev.	Perm.	Rev.	Perm.
0.0001	0	0	0	0	0	0	57	41	0	0
0.001	0	0	0	0	5	5	65	52	0	1
0.01	3	3	3	2	19	34	74	62	15	15
0.1	34	24	24	20	155	173	83	75	175	138
1	318	223	374	194	NR	NR	91	91	1074	1019

**Table 3.** Running times in seconds and number of matches for IMPALA and BLASTP using wolf1187 PSSMs and *M.genitalium* queries. Protein sequences are identified using NCBI gi numbers

Sequence	Length	IMPALA time	BLASTP time	IMPALA matches	BLASTP matches
1350726	48	4	24	1	39
1350690	100	6	32	6	35
1351488	200	10	41	5	21
1346397	297	16	54	23	135
1351482	398	19	73	7	60
1346141	508	24	88	12	79
1709636	607	32	98	27	98
1351503	701	33	116	15	214
1351145	900	43	152	16	169
1351473	1206	55	182	12	112

UltraSparc processor of a Sun Ultra Enterprise 4000/5000 server with 768 MBytes of RAM. This computer runs the operating system Solaris, version 2.6 which is an implementation of UNIX. We used the current Sun C compiler, with optimization turned on, for both impala and BLASTP. The times in the table are the sum of user and system times reported by the time command for the faster of two identical runs. All times are in seconds.

Table 3 shows that IMPALA requires approximately 4 extra seconds for every additional 100 residues, and a little extra time when there are more matches.

While conducting the sensitivity comparison of IMPALA and HMMER described above, we also measured running time. The total time for all runs of the impala program was 30 264 s; the total time for all the runs of the hmmpfam program was 39 471 s. This comparison does not include preprocessing time because most of the work to prepare the inputs to hmmpfam had already been done as part of the ongoing SMART project (Ponting *et al.*, 1999b) and related research.

## Discussion

IMPALA allows rapid, rigorous searching of databases of PSSMs created by PSI-BLAST. Other combinations

of PSSM databases and search tools exist, including Pfam/HMMER (Eddy, 1996; Sonnhammer *et al.*, 1997; Bateman *et al.*, 1999), SMART/HMMER (Eddy, 1996; Ponting *et al.*, 1999b), PROSITE/pfscan (Bucher *et al.*, 1996), but none of these allows searching of PSI-BLAST created PSSMs. At the recent CASP3 protein structure prediction competition many of the presenters used PSI-BLAST to construct their PSSMs (Sternberg *et al.*, 1999), which suggests that a search tool compatible with PSI-BLAST PSSMs is needed.

IMPALA can be used with multiple (arbitrary) databases, not a specific database like Pfam or SMART. We have demonstrated that IMPALA gives accurate predictions, is statistically accurate, and fast.

IMPALA's algorithmic methods may be used with PSSMs generated by other programs. For syntactic reasons, however, the software is strongly tied to PSI-BLAST. The PSSMs in the library are assumed to be in PSI-BLAST byte-encoded format. Each PSSM is assumed to have an associated protein sequence to define its columns. This allows BLAST output code to display alignments between IMPALA's query sequences and the representative sequences of any PSSMs that they match. Therefore, most existing programs for post-processing

BLAST output should be able to parse IMPALA output as well. This design decision was validated when IMPALA was recently added as a search option to the Blocks Database Server, which already had a PSI-BLAST option. Because IMPALA produces output that is syntactically very similar to BLAST output, much of the WWW-related code could be reused in installing an IMPALA option.

Because IMPALA solves some of the same subproblems as BLASTP, but is a small and malleable piece of software, it is a useful testbed for prototyping improvements to BLASTP. For example, we plan to reuse the IMPALA code for more precise alignment scoring and statistical assessment in trial versions of PSI-BLAST.

The IMPALA source code, the wolf1187 database, and the aravind105 database are freely available from the NCBI ftp site [ncbi.nlm.nih.gov](http://ncbi.nlm.nih.gov). The two databases may be found in the subdirectory [ftp://ncbi.nlm.nih.gov/pub/impala](http://ncbi.nlm.nih.gov/pub/impala). The source code is in [ftp://ncbi.nlm.nih.gov/toolbox/ncbi\\_tools](http://ncbi.nlm.nih.gov/toolbox/ncbi_tools), and some IMPALA executables for different implementations of UNIX are in [ftp://ncbi.nlm.nih.gov/blast/executables](http://ncbi.nlm.nih.gov/blast/executables). IMPALA has recently been added as a search option on the Blocks Database Server (<http://blocks.fhcrc.org/blocks/impala.html>) using a different library of PSSMs derived from the BLOCKS database (Henikoff and Henikoff, 1994). The databases are distributed as a collection of ASCII files computed by the first preprocessor program makemat, ready for input to the second preprocessor program copymat. The user needs to run copymat just once for each database.

## Acknowledgements

We thank Dr David Lipman for stimulating discussions on the IMPALA project. We thank Dr Jorja Henikoff for including IMPALA as an option on the Blocks Database Server. We thank Drs Dmitrij Frishman, Jorja Henikoff, Steven Henikoff, and Anna Panchenko for testing IMPALA and providing useful comments.

## References

- Altschul,S.F. and Gish,W. (1996) Local alignment statistics. *Methods Enzymol.*, **266**, 460–480.
- Altschul,S.F. and Koonin,E.V. (1998) Iterated profile searches with PSI-BLAST – a tool for discovery in protein databases. *Trends Biochem. Sci.*, **23**, 444–447.
- Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.
- Altschul,S.F., Boguski,M.S., Gish,W. and Wootton,J.C. (1994) Issues in searching molecular databases. *Nat. Genet.*, **6**, 119–129.
- Altschul,S.F., Madden,T.L., Schäffer,A.A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.
- Aravind,L. and Koonin,E.V. (1999) Gleaning non-trivial structural, functional and evolutionary information about proteins by iterative database searches. *J. Mol. Biol.*, **287**, 1023–1040.
- Bailey,T.L. and Elkan,C. (1995) The value of prior knowledge in discovering motifs with MEME. In Rawlings,C., Clark,D., Altman,R., Hunter,L., Lengauer,T. and Wodak,S. (eds), *Proceedings of the Third International Conference Intelligent Systems for Molecular Biology* AAAI Press, Menlo Park, California, pp. 21–29.
- Bailey,T.L. and Gribskov,M. (1998) Combining evidence using p-values: application to sequence homology searches. *Bioinformatics*, **14**, 48–54.
- Baldi,P. and Chauvin,Y. (1994) Hidden Markov models of the G-protein-coupled receptor family. *J. Comput. Biol.*, **1**, 311–336.
- Baldi,P., Chauvin,Y., Hunkapiller,T. and McClure,M.A. (1994) Hidden Markov models of biological primary sequence information. *Proc. Natl. Acad. Sci. USA*, **91**, 1059–1063.
- Bateman,A., Birney,E., Durbin,R., Eddy,S.R., Finn,R.D. and Sonnhammer,E.L.L. (1999) Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins. *Nucleic Acids Res.*, **27**, 260–262.
- Berg,O.G. and von Hippel,P.H. (1987) Selection of DNA binding sites by regulatory proteins. Statistical-mechanical theory and application to operators and promoters. *J. Mol. Biol.*, **193**, 723–750.
- Brenner,S.E., Chothia,C. and Hubbard,T.J. (1998) Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl. Acad. Sci. USA*, **95**, 6073–6078.
- Brown,M., Hughey,R., Krogh,A., Mian,I.S., Sjölander,K. and Haussler,D. (1993) Using Dirichlet mixture priors to derive hidden Markov models for protein families. In Hunter,L., Searls,D. and Shavlik,J. (eds), *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology* AAAI Press, Menlo Park, California, pp. 47–55.
- Bucher,P., Karplus,K., Moeri,N. and Hofmann,K. (1996) A flexible motif search technique based on generalized profiles. *Comput. Chem.*, **20**, 3–23.
- Chervitz,S.A., Aravind,L., Sherlock,G., Ball,C.A., Koonin,E.V., Dwight,S.S., Harris,M.A., Dolinski,K., Mohr,S., Smith,T., Weng,S., Cherry,J.M. and Botstein,D. (1998) Comparison of the complete protein sets of worm and yeast: orthology and divergence. *Science*, **282**, 2022–2028.
- Collins,J.F., Coulson,A.F.W. and Lyall,A. (1988) The significance of protein sequence similarities. *Comput. Appl. Biosci.*, **4**, 67–71.
- Dembo,A., Karlin,S. and Zeitouni,O. (1994) Limit distribution of maximal non-aligned two-sequence segmental score. *Ann. Prob.*, **22**, 2022–2039.
- Dodd,I.B. and Egan,J.B. (1987) Systematic method for the detection of potential lambda cro-like DNA-binding regions in proteins. *J. Mol. Biol.*, **194**, 557–564.
- Durbin,R., Eddy,S., Krogh,A. and Mitchison,G. (1998) *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge.
- Eddy,S.R. (1996) Hidden Markov models. *Curr. Opin. Struct. Biol.*, **6**, 361–365.
- Eddy,S.R. (1998) Profile hidden Markov models. *Bioinformatics*, **14**, 755–763.
- Fischer,D. and Eisenberg,D. (1997) Assigning folds to the proteins

- encoded by the genome of *Mycoplasma genitalium*. *Proc. Natl. Acad. Sci. USA*, **94**, 11929–11934.
- Fraser, C.M., Gocayne, J.D., White, O., Adams, M.D., Clayton, R.A., Fleischmann, R.D., Bult, C.J., Kerlavage, A.R., Sutton, G., Kelley, J.M., Fritchman, J.L., Weidman, J.F., Small, K.V., Sandusky, M., Fuhrmann, J., Nguyen, D., Utterback, T.R., Saudek, D.M., Phillips, C.A., Merrick, J.M., Tomb, J.F., Dougherty, B.A., Bott, K.F., Hu, P.C., Lucier, T.S., Peterson, S.N., Smith, H.O., Hutchison, C.A. and Venter, J.C. (1995) The minimal gene complement of *Mycoplasma genitalium*. *Science*, **270**, 397–403.
- Gish, W. and States, D.J. (1993) Identification of protein coding regions by database similarity search. *Nat. Genet.*, **3**, 266–272.
- Gotoh, O. (1982) An improved algorithm for matching biological sequences. *J. Mol. Biol.*, **162**, 705–708.
- Gribskov, M., McLachlan, A.D. and Eisenberg, D. (1987) Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. USA*, **84**, 4355–4358.
- Gribskov, M., Lüthy, R. and Eisenberg, D. (1990) Profile analysis. *Methods Enzymol.*, **183**, 146–159.
- Henikoff, S. and Henikoff, J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, **89**, 10915–10919.
- Henikoff, S. and Henikoff, J.G. (1994) Protein family classification based on searching a database of blocks. *Genomics*, **19**, 97–107.
- Henikoff, J.G., Henikoff, S. and Pietrokovski, S. (1999) New features of the blocks database servers. *Nucleic Acids Res.*, **27**, 226–228.
- Hofmann, K., Bücher, P., Falquet, L. and Bairoch, A. (1999) The PROSITE database, its status in 1999. *Nucleic Acids Res.*, **27**, 215–219.
- Hubbard, T.J.P., Ailey, B., Brenner, S.E., Murzin, A.G. and Chothia, C. (1999) SCOP: a structural classification of proteins database. *Nucleic Acids Res.*, **27**, 254–256.
- Huynen, M., Doerks, T., Eisenhaber, F., Orengo, C., Sunyaev, S., Yuan, Y. and Bork, P. (1998) Homology-based fold predictions for *Mycoplasma genitalium* proteins. *J. Mol. Biol.*, **280**, 323–326.
- Karlin, S. and Altschul, S.F. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, **87**, 2264–2268.
- Krogh, A., Brown, M., Mian, I.S., Sjölander, K. and Haussler, D. (1994) Hidden Markov models in computational biology. Applications to protein modeling. *J. Mol. Biol.*, **235**, 1501–1531.
- Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F. and Wootton, J.C. (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, **262**, 208–214.
- McLachlan, A.D. (1983) Analysis of gene duplication repeats in the myosin rod. *J. Mol. Biol.*, **169**, 15–30.
- Mott, R. (1992) Maximum-likelihood estimation of the statistical distribution of Smith–Waterman local sequence similarity scores. *Bull. Math. Biol.*, **54**, 59–75.
- Murzin, A.G., Brenner, S.E., Hubbard, T. and Chothia, C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**, 536–540.
- Mushegian, A.R., Bassett, D.E. Jr, Boguski, M.S., Bork, P. and Koonin, E.V. (1997) Positionally cloned human disease genes: patterns of evolutionary conservation and functional motifs. *Proc. Natl. Acad. Sci. USA*, **94**, 5831–5836.
- Neuwald, A.F., Liu, J.S., Lipman, D.J. and Lawrence, C.E. (1997) Extracting protein alignment models from the sequence database. *Nucleic Acids Res.*, **25**, 1665–1677.
- Park, J., Karplus, K., Barrett, C., Hughey, R., Haussler, D., Hubbard, T. and Chothia, C. (1998) Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J. Mol. Biol.*, **284**, 1201–1210.
- Pathy, L. (1987) Detecting homology of distantly related proteins with consensus sequences. *J. Mol. Biol.*, **198**, 567–577.
- Pearson, W.R. (1998) Empirical statistical estimates for sequence similarity searches. *J. Mol. Biol.*, **276**, 71–84.
- Pearson, W.R. and Lipman, D.J. (1988) Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, **85**, 2444–2448.
- Ponting, C.P., Aravind, L., Schultz, J., Bork, P. and Koonin, E.V. (1999a) Eukaryotic signalling domain homologues in archaea and bacteria. Ancient ancestry and horizontal gene transfer. *J. Mol. Biol.*, **289**, 729–745.
- Ponting, C.P., Schultz, J., Milpetz, F. and Bork, P. (1999b) SMART: identification and annotation of domains from signalling and extracellular protein sequences. *Nucleic Acids Res.*, **27**, 229–232.
- Rychlewski, L., Zhang, B. and Godzik, A. (1998) Fold and function predictions for *Mycoplasma genitalium* proteins. *Fold. Des.*, **3**, 229–238.
- Schneider, T.S., Stormo, G.D., Gold, L. and Ehrenfeucht, A. (1986) Information content of binding sites on nucleotide sequences. *J. Mol. Biol.*, **188**, 415–431.
- Smith, T.F. and Waterman, M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.
- Smith, T.F., Waterman, M.S. and Burks, C. (1985) The statistical distribution of nucleic acid similarities. *Nucleic Acids Res.*, **13**, 645–656.
- Sonnhammer, E.L., Eddy, S.R. and Durbin, R. (1997) Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins*, **28**, 405–420.
- Staden, R. (1984) Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res.*, **12**, 505–519.
- Sternberg, M.J.E., Bates, P.A., Kelley, L.A. and MacCallum, R.M. (1999) Progress in protein structure prediction: assessment of CASP3. *Curr. Opin. Struct. Biol.*, **9**, 368–373.
- Stormo, G.D. and Hartzell, G.W. III (1989) Identifying protein-binding sites from unaligned DNA fragments. *Proc. Natl. Acad. Sci. USA*, **86**, 1183–1187.
- Tanaka, H., Ishikawa, M., Asai, K. and Konagawa, A. (1993) Hidden Markov models and iterative aligners: study of their equivalence and possibilities. In Hunter, L., Searls, D. and Shavlik, J. (eds), *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology* AAAI Press, Menlo Park, CA, pp. 395–401.
- Tatusov, R.L., Altschul, S.F. and Koonin, E.V. (1994) Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proc. Natl. Acad. Sci. USA*, **91**, 12091–12095.
- Taylor, W.R. (1986) Identification of protein sequence homology by consensus template alignment. *J. Mol. Biol.*, **188**, 233–258.
- Teichmann, S.A., Park, J. and Chothia, C. (1998) Structural assign-

- ments to the *Mycoplasma genitalium* proteins show extensive gene duplications and domain rearrangements. *Proc. Natl. Acad. Sci. USA*, **95**, 14658–14663.
- Teichmann,S.A., Chothia,C. and Gerstein,M. (1999) Advances in structural genomics. *Curr. Opin. Struct. Biol.*, **9**, 390–399.
- Wallace,J.C. and Henikoff,S. (1992) PATMAT: a searching and extraction program for sequence, pattern and block queries and databases. *Comput. Appl. Biosci.*, **8**, 249–254.
- Waterman,M.S. and Eggert,M. (1987) A new algorithm for best subsequence alignments with application to tRNA–rRNA comparisons. *J. Mol. Biol.*, **197**, 723–728.
- Waterman,M.S. and Vingron,M. (1994) Rapid and accurate estimates of statistical significance for sequence database searches. *Proc. Natl. Acad. Sci. USA*, **91**, 4625–4628.
- Wolf,Y.I., Brenner,S.E., Bash,P.A. and Koonin,E.V. (1999) Distribution of protein folds in the three superkingdoms of life. *Genome Res.*, **9**, 17–26.
- Wootton,J.C. and Federhen,S. (1993) Statistics of local complexity in amino acid sequences and sequence databases. *Comput. Chem.*, **17**, 149–163.
- Yi,T.-M. and Lander,E.S. (1994) Recognition of related proteins by iterative template refinement (ITR). *Protein Sci.*, **3**, 1315–1328.
- Zhang,Z., Berman,P. and Miller,W. (1998) Alignments without low-scoring regions. *J. Comput. Biol.*, **5**, 197–210.